1 Описание

Игрой называется идеализированная математическая модель конфликтной ситуации. Стороны, участвующие в конфликте называются игроками, а исход конфликта – выигрышем.

Игра состоит из последовательности действий (ходов), которые подразделяются на личные (совершаемые игроками осмысленно на основе некоторого правила – стратегии) и случайные (не зависящие от игроков). В теории игр рассматриваются ситуации, в которых обязательно присутствуют личные ходы.

Игроки - стороны, участвующие в игре: 2 игрока - парная игра, более 2 - множественная игра. Игра с нулевой суммой (антагонистическая игра) - выигрыш одного из игроков равен проигрышу другого. Антагонистическая игра называется матричной, если множества стратегий игроков конечны: $X = 1, ...m, Y = 1, ..., n$. При этом принято обозначать стратегию первого игрока через i, стратегию второго через j, а выигрыш первого (проигрыш второго) $F(i, j)$ через a_{ij}. Матрица $A = (a_{ij})_m \times_n$ называется матрицей игры. Первый игрок выбирает в ней номер строки i, а второй номер столбца j.

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & ... & a_{1n} \\ a_{21} & a_{22} & a_{23} & ... & a_{2n} \\ ... & ... & ... & ... & ... \\ a_{m1} & a_{m2} & a_{m3} & ... & a_{mn} \end{bmatrix}$$

Первый игрок получает максимальный гарантированный (не зависящий от поведения второго игрока) выигрыш, равный цене игры, аналогично, второй игрок добивается минимального гарантированного проигрыша.

Решение антагонистической игры - выбор стратегий, удовлетворяющих условию оптимальности (игрок А должен получать максимальный выигрыш, когда игрок В придерживается своей стратегии, а игрок В должен получать минимальный проигрыш, когда игрок А придерживается своей стратегии).

Стратегия игрока – это набор правил, используемых при выборе очередного личного хода. Целью игры является нахождение оптимальной стратегии для каждого игрока, т. е. такой, при которой достигается максимум ожидаемого среднего выигрыша при многократном повторении игры.

Чистая стратегия даёт полную определённость, каким образом игрок продолжит игру. В частности, она определяет результат для каждого возможного выбора, который игроку может придётся сделать. Пространством стратегий называют множество всех чистых стратегий, доступных данному игроку.

Смешанная стратегия является указанием вероятности каждой чистой стратегии.

Седловой точкой матрицы называется такая пара номеров строки и столбца, что для любых $i = (1, \ldots, m)$ и $j = (1, \ldots, n)$ выполняются неравенства $a_{ij_0} \leq a_{i_0j_0} \leq a_{i_0j}$. Таким образом, элемент $a_{i_0j_0}$ в матрице с седловой точкой (i_0, j_0) является масксимальным в своем столбце и минимальным в своей строке, что и объясняет нецелесообразность выбора любым из игроков другой стратегии.

2 Математическое решение

Для поиска оптимальных стратегий и значения игры сведем ее к к паре двойственных задач линейного программирования (ЛП). В последствии, решим их при помощи симплекс-метода.

Общей задачей линейного программирования называется задача, которая состоит в определении максимального (минимального) значения функции

$$F = \sum_{j=1}^{n} c_j x_j$$

при условии

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_j, (i = \overline{1, k}),$$

$$\sum_{j=1}^{n} a_{ij} x_j = b_j, (i = \overline{k+1, m}),$$

$$x_j \geq 0, (j = \overline{1, l}, l \leq n),$$

где a_{ij}, b_i, c_j - заданные постоянные величины и $k \leq m$.

Канонической задачей линейного программирования называется задача, в которой требуется найти максимум целевой функции при ограничениях, заданных системой линейных уравнений.

$$L(X) = \sum_{j=1}^{n} c_j x_j \to max$$

$$\begin{cases} \sum\limits_{j=1}^{n} a_{ij} = b_i, \forall i \in N_m \\ x_j \geq 0, \forall j \in N_n \end{cases}$$

Общая двойственная задача — это задача симметричная общей прямой задаче линейного программирования, с ограничениями всех видов и с переменными всех видов.

$$D(Y) = \sum_{i=1}^{m} b_i x_i \to min$$

$$\begin{cases} \sum\limits_{i=1}^{m} a_{ij} y_i \geq c_j, \forall j \in N_{n_1} \\ \sum\limits_{i=1}^{m} a_{ij} y_i = c_j, \forall j \in N_{n_2} \backslash N_{n_1} \\ \sum\limits_{i=1}^{m} a_{ij} y_i \leq c_j, \forall j \in N_n \backslash N_{n_2} \\ y_i \geq 0, \forall i \in N_{m_1} \\ y_i \leq 0, \forall i \in N_m \backslash N_{m_2} \\ y_i \in R, \forall i \in N_m \end{cases}$$

Задачей линейного программирования в стандартной (нормальной) форме, называется задача, в которой требуется найти максимум целевой функции при ограничениях, заданных системой неравенств одного смысла.

Если все или некоторые ограничения в системе заданы неравенствами, то задачу можно свести к канонической путём преобразования неравенств в уравнения.

Симплекс метод - это метод последовательного перехода от одного базисного решения системы ограничений задачи линейного программирования к другому базисному решению до тех пор, пока функция цели не примет оптимального значения (максимума или минимума). Алгоритм:

1. Привести задачу к каноническому виду, заменяя все неравенства на уравнения с помощью введения новых переменных.

2. Записать систему уравнений соответсвующую данной задаче:

$$\begin{cases} z - \sum\limits_{i=1}^{n} c_i x_i = 0 \\ \sum\limits_{j=1}^{n} a_{ij} x_j = b_i, i = 1, ..., m \end{cases}$$

3. Привести систему к диагональному виду по базисным переменным

$$z, x_1, ..., x_m.$$

4. Составить таблицу коэффициентов:

	z	x_1	...	x_m	x_{m+1}	...	x_n
z	z_0	0	...	0	c_{m+1}	...	c_n
x_1	b_1	1	...	0	$a_{1,m+1}$	...	a_{1n}
...	...	...	...	...	...	...	...
x_m	b_m	0	...	1	$a_{m,m+1}$	...	a_{mn}

5. Если все оэффициенты в первой строке при небазисных переменных $x_{m+1}, ..., x_n$ неотрицательны, то текущее базисное решение - оптимальное. Если это не так, то в число базисных переменных вводим переменную x_s, где номер s такой, что $c_s = \min\limits_{c_j<0} c_j$.

6. Столбец с номером s - ведущий. Если в нем нет положительных коэффициентов, то оптимального решения не существует. В противном случае в качестве базисной переменной, которая исключается из числа базисных, выбирается та переменная x_r, для которой

$$\frac{b_r}{a_{rs}} = \min_{a_{is}>0} \frac{b_i}{a_{is}}$$

7. a_{rs} называется ведущим элементом. Используя эквивалентные преобразования таблицы, пересичтываем таблицу так, чтобы ведущий элемент стал равным 1, а все остальные элементы ведущего столбца - равным 0.

8. Перейти к пункту 5 для исследования получившейся таблицы.

3 Реализация программы

На вход функции $nash\_equilibrium()$ подаётся матрица $m \times n$, где m - число статегий первого игрока, n соотвественно второго игрока.С помо-

щью функции $shape()$ в переменную str заносим число строк, а в переменную col число столбцов.Далее, с помощью $min()$ находим минимальный элемент матрицы. Если этот элемент отрицательный, тогда из матрицы вычитаем $(-min\_element + 1)$ чтобы предотвратить деление на 0.

С помощью функции $linprog()$ из библиотеки SciPy в дальнейшем решаем задачу ЛП. Задача функции $linprog()$ - это решение задачи ЛП.

$linprog(c, A\_ub = None, b\_ub = None, A\_eq = None, b\_eq = None, bounds = None, method =' simplex', callback = None, options = None)$

x = linprog(c,A,b) находит $\min\limits_{x} c^T * x$ при условии, что :

$$A * x \leq b$$

$$A\_eq * x == b\_eq$$

Приводим матрицу и векторы c и b к каноническому виду для функции $linprog()$. Затем ищем оптимальную стратегию второго игрока str2, для этого мы подаем на вход функции $linprog()$ вектор c, матрицу игры A и вектор b, для первого игрока аналогично, за исключением того, что матрицу мы транспонируем (функцией $transpose()$ из библиотеки NumpPy) и берем со знаком минус. Далее получаем через .x векторы оптимальной стратегии, а затем узнаем цену игры game, поделив единицу на суммарное значение всех элементов вектора str2. Функция $nash\_equilibrium()$ возвращает цену игры, а также векторы str1 и str2.

Для вывода ответа в виде рациональных дробей используем функцию frac(game,str1,str2). В которой мы применяем метод $as\_integer\_ratio()$, а затем $limit\_denominator()$, тем самым приводя дробь к нормальному виду.

Для визуализации спектров оптимальных стратегий игроков вызывается функция plot(p, q). Ее мы реализуем с помощью функций:

$plt.figure()$ с параметром figsize, которая создает область рисования 10 на 10.

$fig.add\_subplot(pos)$, чтобы добавить оси к фигуре как часть схемы подзаголовка. Аргумент - трехзначное целое число, первая цифра – кол-во строк, вторая – кол-во столбцов, а третья - индекс подзаголовка.

$Stem()$ визуализирует спектры, отображая вертикальные линии в каждом месте от базовой линии до длины вектора

4 Используемые модули

В программе были использованы следующие модули:

•Scipy

SciPy - это основанная на Python экосистема программного обеспечения с открытым исходным кодом для математики, науки и техники. В нашей программе мы используем функцию linprog() из пакета scipy.optimize.

•NumPy

NumPy - это open-source модуль для Python, который предоставляет общие математические и числовые операции в виде пре-скомпилированных, быстрых функций. В нашем случае, мы используем NumPy для создания и поддержки матриц, вместе с операциями над ними.

•Matplotlib.pyplot

Matplotlib.pyplot - это модуль для Python для визуализации данных двухмерной графикой (трёхмерная графика поддерживается). С помощью него мы строим графики.

•Fractions

Fractions - это модуль предоставляет поддержку рациональных чисел. С его помощью мы представляем полученные в программе рациональные числа в виде дробей.